Global solar radiation time series forecasting using different architectures of the multilayer perceptron model

Global solar radiation time series forecasting using different architectures of the multilayer perceptron model

J J Peñalva[1*], D A Lozano[2], J C Murillo[2] and F M Ortega[3]

[1]Faculty of Mechanical Engineering, Universidad Nacional de Ingeniería, Lima, Peru
[2]Faculty of Informatics Engineering, Pontificia Universidad Católica del Perú, Lima, Peru
[3]Faculty of Chemical Engineering, Universidad Nacional del Callao, Lima, Peru

*E-mail: jpenalvas@uni.pe

Abstract. In this work, the multilayer perceptron model was used to forecast the time series of global solar radiation for a near future about a week. Different architectures of this model were built through varying its different hyperparameters such as optimizers, activation functions, number of neurons and neuron dropout in which their performance was evaluated using error metrics. It was found that the architectures (60, SGD, Sigmoid), (10, Adam, Relu) and (60, SGD, Sigmoid) presented an R^2 around 0.877, 0.873 and 0.872, respectively. The architecture with neuron dropout (150, SGD, Sigmoid, 0.2) presented a higher performance among all the architectures evaluated and its R^2 value was 0.884. Architectures with higher performance are used to predict future values of solar radiation.

1. Introduction

In recent years, energy demand has increased due to industrial development, the growth of the economy and population [1]. This high energy demand produces a large increase in the burning of fossil fuels that releases large amounts of carbon dioxide and toxic gases into the atmosphere causing climate change and global warming, making it a global concern because of change on normal conditions of planet [2]. To mitigate and solve this problem, the use of clean energy such as renewable energy was increased to decarbonize the conventional electricity grid [3].

Solar energy is one of the most promising sources due to its availability and energy potential [4]. For this reason, the generation of photovoltaic energy has been increasing its installation both for systems connected and isolated from the electricity grid [5]. However, renewable energy such as solar and wind present an intermittent generation due to its variable nature caused by climatic factors [6], making it necessary to enter other generators and storage systems such as diesel and batteries to satisfy the energy demand of the place in a reliable way [7, 8].

In order to ensure the precise balance between production and electricity consumption using photovoltaic energy with greater penetration in the electrical grid, the precise forecasting of the solar resource is important to solve the problem of its uncertainty [9]. This solution allows the grid operator to guarantee a precise balance with a greater share of photovoltaic generation in a safe and reliable way [10-12].

In the literature, there are a lot of ways for forecasting time series such as classic models of the type Exponential Smoothing, Holt-Winters, Box-Jenking [13-14] and intelligent models such multilayer

perceptron neural network [15]. Multilayer perceptron performance (MLP) is dependent on many hyperparameters, its evaluating is important to get the best model predictive [16].

This study aims to evaluate the combination of these hyperparameters and have good model performances. The work is organized as follows: Section 1 shows the introduction. Section 2 describes the methodology. Section 3 presents the results and discussions of different architectures of the MLP model. In section 4, we conclude and suggest future works.

2. Methodology

In this work, the solar radiation time series is used to predict future values using different configurations of the multilayer perceptron model. Time series data of global solar radiation is obtained NASA' website for a study location, which is used to train a supervised learning model and predict its future behavior.

The flow diagram of the data processing, configuration, evaluation and forecasting of the MLP model is shown in figure 1. The first stage is about the data preparation process, where the data is divided into training and test data. Training data is normalized or scaled between values from 0 to 1 using the MinMax method [17]. Within this stage, an important sub-process is the transformation of the training data into input data for the MLP model, this transformation is carried out using the sliding window method that allows generating data series displaced at a later time [18-19].

The second stage is the configuration and training of the MLP model. In this stage, the model is defined by different architectures due to its hyperparameters. Each MLP architecture uses data to train and find a minimum error in its predictions.

The evaluation of the MLP model is the third stage, where the predictions of each architecture are evaluated with test data through error metrics, which indicates the performance of each architecture. High performance architectures continue to the next stage, otherwise they are trained again. Finally, the last stage is the prediction of future values or forecast. At this stage, the highest performing architectures are saved and used to predict future values of time series.

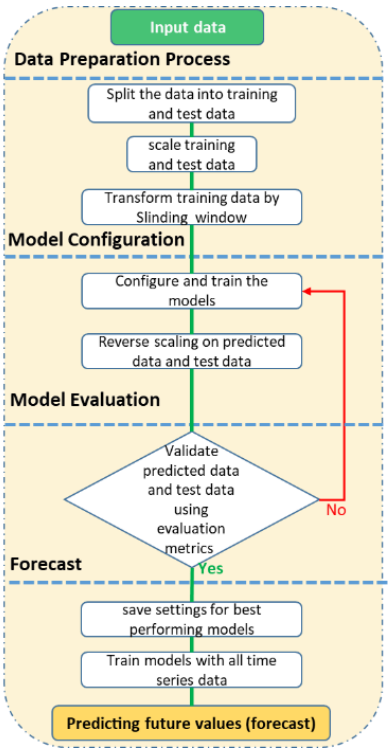

Figure 1. Flow chart to obtain future values: Data preparation process, configuration, evaluation and forecast of the model.

2.1. Study area and data of solar radiation

The study place was chosen in the community called San Juan de Tarucani which is located in Peru, Arequipa region with coordinates 16° 11' S and 71° 3.6' E of latitude and longitude, respectively.

Time series data of global solar radiation obtained at these coordinates is shown in figure 2a. On the other hand, in figure 2b, the radiation curve in hours and months is shown in a 3-dimensional graph; This curve shows greater radiation in the summer months, reaching values of 1.2 kW/m^2.

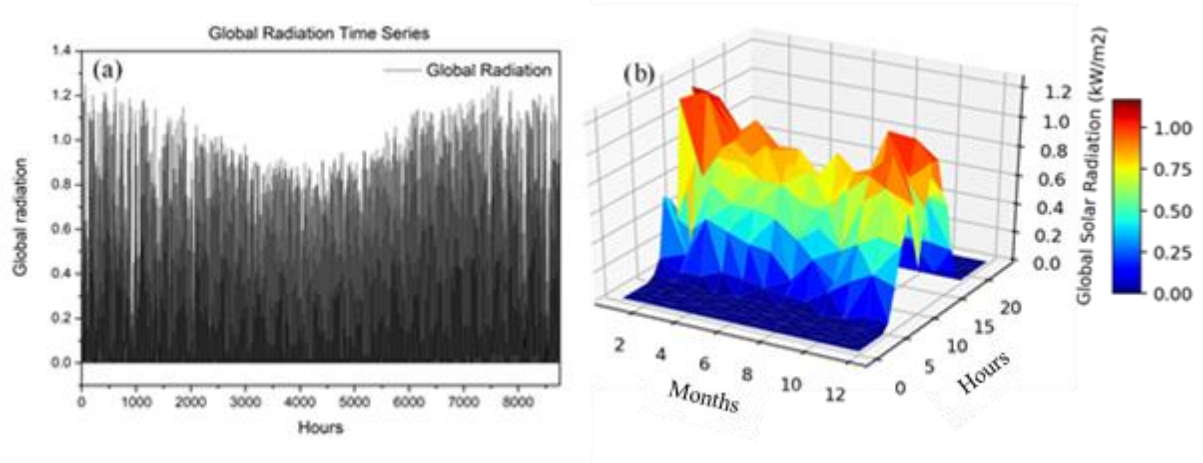

Figure 2. (a) Time series of solar radiation from the study site throughout the year. (b) Graph of global solar radiation in 3D for months and hours.

2.2. Data preparation process: Data partitioning, scaling and transformation

The time series data is partitioned into training and test data with 90% and 10%, respectively. The training data allows to find the optimal weights of the MLP model to ensure efficient learning and high performance in the predictions on training and test data.

Intelligent models such as MLP improve their performance when their input values are scaled in a standard range. The MinMax method is used to scale data values between the range of 0 and 1. The training data is scaled to train MLP model through equation 1, and the predicted values by the model are rescaled to their normal values through the equation 2:

$$x_i' = \frac{x_i - \min(x_i)}{\max(x_i) - \min(x_i)} \tag{1}$$

$$x_i = x_i'(\max(x_i) - \min(x_i)) + \min(x_i) \tag{2}$$

Where min (x_i) and max (x_i) are the minimum and maximum values that are present in the training data, respectively, x_i' is the scaled value of x_i which is between values of 0 and 1.

Univariate time series are a unique set of observable values in a time order that can be predicted using supervised learning models such as the MLP model. However, this set of time series requires being prepared as input and output data to train the model. This preparation consists of generating samples with observable values shifted in previous time steps as input variable and an observable value of next time step as output variable. This procedure is called the sliding window method, where the number of previous time steps is called the window width, and each window shift predicts an observable value.

Figure 3 graphically shows data preparation, training, and prediction of the MLP model. The value of (t + N) and M represent the number of observable values for training and test data, respectively. When preparing data, the observable values of the time series were divided into (N + M + 1) samples with t number of time steps as input variable and one time step as output variable. The new training data has (N + 1) sliding windows or samples which are used as input data to train the MLP model and predict the

following values. When the model finds an optimal learning with new training data and a minimum error, the predictions on test data are carried out. These predicted values are obtained when test data samples are entered into the model; As an example, when entering the sample with label (N + 2) to the model, an observable value $x_{(t+N+2)}$ is obtained. These predictions are evaluated with values of test data using error metrics.

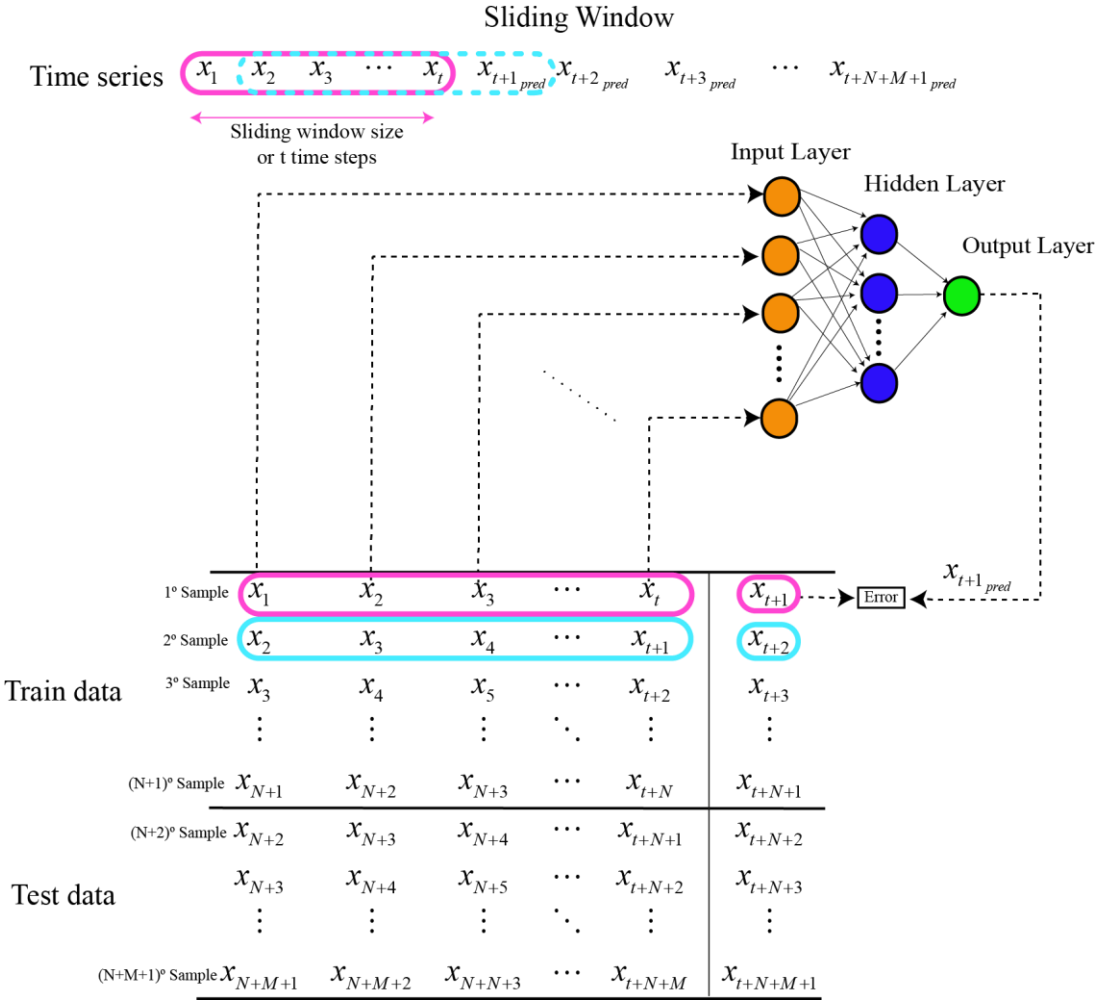

Figure 3. Data preparation, training and prediction of the MLP model.

2.3. The multilayer perceptron (MLP) model

Within the artificial neural network, the MLP model is one of the most used architectures because it is an efficient and practical learning algorithm, which allows a wide application in various fields of science and industry [20-22].

Multilayer perceptron neural network is comprised of three types of layers, which are input layer, hidden layer and output layer where each one has a certain number of neurons which are interconnected. The data processing of the MLP model has several stages [23-24]: In the first stage, this process occurs in the input layer neurons and hidden layer neurons. The observable values assigned to each neuron in the input layer are multiplied by the assigned weights, then added by bias and stored in each neuron in the hidden layer. This process is mathematically expressed in equation (3):

$$h_l = \sum_{i=1}^{t} w_{il} x_i + \beta_l \quad , \forall l \in \{1, 2, 3, ..., j\} \tag{3}$$

Where l represents the number of input neurons, x_i represents the values of each neuron in the input layer and w_{il} is the weight assigned to connect the input neuron x_i and the hidden neuron l. The second term is the bias β_l which is assigned to each neuron in the hidden layer. The sum of the product between input values and their assigned weights, plus the bias of each hidden neuron are represented by h_l.

The second stage occurs at the output of hidden layer neurons where the output values of each hidden neuron are calculated as follows:

$$H_l = f(h_l), \forall l \in \{1,2,3,...,j\} \tag{4}$$

Where H_l is the output value of each hidden neuron and f is the activation function that transforms the values of h_l.

The activation functions are varied, being the most used the types of activation functions Sigmoid, Tanh and Relu [25], which are represented by equations (5), (6) and (7), respectively:

$$f(h_l) = \frac{1}{1+e^{-h_l}} \tag{5}$$

$$f(h_l) = \frac{(e^{h_l} - e^{-h_l})}{(e^{h_l} + e^{-h_l})} \tag{6}$$

$$f(h_l) = \begin{cases} 0, & if\ h_l < 0 \\ h_l, & if\ h_l \geq 0 \end{cases} \tag{7}$$

The last stage occurs in the output layer neurons, where the output values of hidden layer neurons are multiplied by new assigned weights and added by a value of bias to each output neuron. That is represented by:

$$O_k = f(\sum_{l=1}^{j} w_{lk} H_l + \beta_k), \forall k \in \{1,2,3,...,n\} \tag{8}$$

Where β_k is the bias of each output neuron, w_{lk} is the weight assigned to the output values of each hidden neuron which is the connection between the hidden neurons and output neurons, f is the activation function of the output layer neurons and O_k is the output value of the multilayer perceptron.

The main objective of the MLP model is to predict values closer to the observable values. For this reason, it is important to minimize the error of the MLP predictions. The cost or error function (E) is represented in equation (9), which has been considered for a single neuron in the output layer making $k = 1$ and $O_1 = y_{pred}$ derived from the equation (8):

$$E = \frac{1}{2}\sum_{i=1}^{n} (y_i - y_{pred})^2 \tag{9}$$

Where y_{pred} is the predicted value by the model and y_i is the observable value of the time series with n observable quantities.

This error function is iteratively minimized by optimization algorithms, which aims to find optimal weights that improve the learning process of the model and minimize the error between predicted and observable values. The MLP model uses several types of optimization algorithms or also called learning algorithms such as Gradient Descending (SGD), RMSprop and Adam, among others [26].

The performance of the MLP model is evaluated using error metrics. These metrics are the mean square error (MSE), mean square error (RMSE), mean absolute error (MAE) and the coefficient of determination (R^2).These metrics indicate how good is the model performs when the MSE, RMSE, and MAE values are close to zero and R^2 is close to 1 [27].

3. Experimental setup of MLP models

To obtain an adequate MLP model, it is important to study its hyperparameters and the performance of its predictions. For this, different hyperparameters are evaluated which generate diverse configurations of the model, among them we have a learning algorithm, activation function, number of neurons in the hidden layer and neuron dropout. There are other hyperparameters such as input windows or input neurons, learning rate, number of output neurons and other parameters which are kept fixed in this study.

The solar radiation time series data is one year of record in units of kW/m^2 per hour and has 8760 observable values that are divided into training and test data in a relationship of 90% and 10%, respectively. In the training data, the minimum and maximum values are 0 and 1.2 kW/m^2 to be used in scaling and rescaling. For data preparation, the observable values of the time series were divided into 7884 samples with 876 number of time steps as input variable and one time step as output variable. These input and output data are used to train the model and make predictions.

Table 1 shows the different values for these hyperparameters used in the MLP configuration, which generate 216 configurations to be evaluated. These configurations are represented as architectures of the type (number of neurons, optimizer, activation function, with and without neuron dropout, among others). The architectures are trained with training data at a learning rate of 0.001 and with 80 training epochs for each optimizer.

The predictions generated by the MLP model architectures are evaluated with the test data using error metrics (MSE, RMSE, MAE, R^2) to indicate how good their performance is. Subsequently, the MLP architectures are saved to later train them with the entire time series and predict future values outside the series.

Table 1. Hyperparameters and their values for various architectures of MLP model.

Hyperparameters	Possible values	Count
Number of time steps or input neurons	876	1
Learning algorithm or optimizer	SGD, Adam, RMSprop	3
Learning rate	0.001	1
Activation function	Sigmoid, Relu and Tanh	3
Hidden layer neurons number	2, 3, 4, 5,6, 10, 20, 40, 60, 80, 100, 150	12
dropping out units	0% (dropout=0) and 20% (dropout=0.2)	2
Number of output time steps or output neurons	1	1
Total number of hyperparameter combinations		216

4. Results and discussion

In this section, the training results, predictions and performance evaluation of the MLP model with different architectures will be given.

4.1. MLP training process for various architectures

The training processes for different MLP architectures are shown in figure 4. These graphs show the learning (solid lines) and validation (dotted lines) convergence curve for each architecture.

Figures 4a - 4c show the overlap of the training and validation curves when neuron dropout is not used (dropout = 0). However, when using neural dropout (dropout = 20%), there is a separation distance between the training and validation curves. Oneto *et al.* [28] indicate that the use of neuron dropout is important to reduce the overfitting generated in training and to improve the performance of the model. Furthermore, in figure 4a and figure 4b, it is observed that the use of neuron dropout and the increase in

the number of hidden neurons decreases the distance between the training and test curve, reaching a minimum error.

Furthermore, these figures show that when neurons in the hidden layer increase for SGD optimizer, activation function (Relu, Tanh, and sigmoid), and neuron dropout, there is an overlap of training curves. The opposite occurs when decreased neurons, these training curves are separated, being observed in figure 4a and figure 4b. It is also observed that the training and validation curves reach a minimum of training loss with the Relu activation function and the SGD optimizer than the other activation functions.

Figure 4d - 4f, the training and validation curves are shown in the same range as the previous curves. These curves converge when neurons are increased except for RMSprop and Tanh, which present a different behavior. It is also observed many pronounced peaks appear as noise in the validation curve being RMSprop and Sigmoid a little more stable.

Figure 4g shows the training and validation curves for the combination of Adam and Relu. Without neuron dropout, curves converge and overlap with little noise; however, when neuron dropout is added, these curves tend to separate with pronounced peaks. On the other hand, in figure 4h and figure 4i, the activation functions (Tanh, sigmoid) with the Adam optimizer have a similar behavior to the curves generated by Tanh and Sigmoid with the RMSprop optimizer.

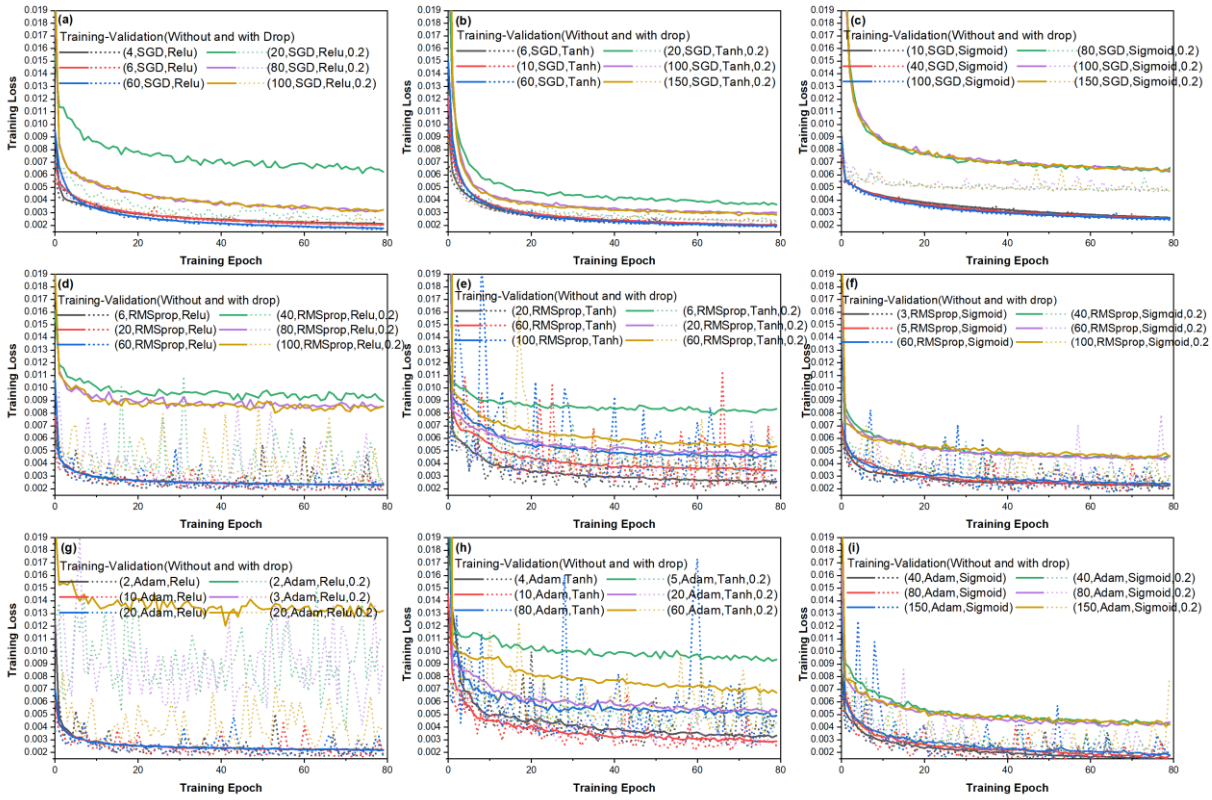

Figure 4. Training and validation curves of the MLP model for different architectures with activation functions (Relu, Tanh, Sigmoid) and the optimizers (SGD, RMSprop, Adam).

4.2. Predictions of different architectures on test data of solar radiation

Figure 5 shows the predictions of global solar radiation for each MLP architecture concerning to its activation function (Relu, Tanh and Sigmoid), optimizer (SGD, RMSprop, Adam), with neuron dropout (0.2) and without neuron dropout (0). The architectures without neuron dropout are shown with solid lines, while the architectures with neuron dropout are represented by dotted lines.

Figure 5a – 5c show the predictions for the architectures (SGD, activation function, dropout=0) with activation functions Relu, Tanh, Sigmoid and different numbers of neurons. These figures show high

and low values of predictions on the peaks of solar radiation profile. It is also observed that the highs and lows of the predictions occur when solar radiation takes zero value.

Figure 5d shows the architecture predictions (RMSprop, Relu, dropout = 0.2, neurons). This figure shows how the predictions of these architectures have good precision with initial observable values and then lose precision. In figure 5 (e) and (f), the predictions improve when considering more neurons.

Figure 5g- 5i, it is observed that the architectures (Adam, Relu, neurons) have better predictions of solar radiation including values close to zero. At these architectures when neuron dropout is considered, the predictions decay over time.

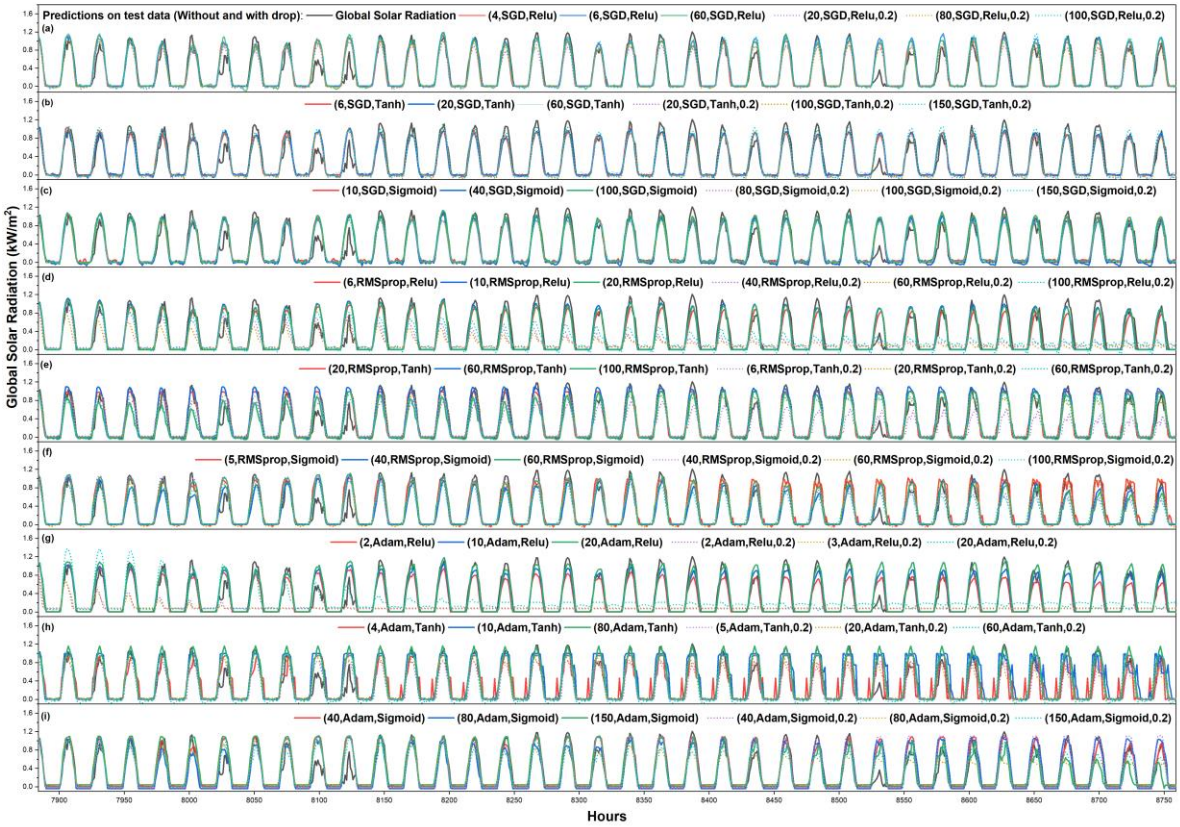

Figure 5. Predictions of the different architectures on the solar radiation test data.

4.3. Error metrics for different MLP architectures

Figure 6 shows the performances of MLP model architectures as a function of neurons number and learning rate (Lr = 0.001), these performances are obtained by means error metrics of type MSE, RMSE, MAE and R^2.

The architectures with SGD optimizer and different activation functions (Relu, Tanh and Sigmoid) have better performance than with the other optimizers, being shown in figure 6a – 6c. On the other hand, in figure 6d - 6f, the architectures with RMSprop optimized and different activation functions have low performance when neurons are increased for cases with and without neuron dropout. It is also observed that the RMSprop optimizer with Tanh and Sigmoid have lower performance than the SGD optimizer and the activation functions studied.

In figure 6g, the architectures with Adam and Relu have similar performances to the RMSprop optimizer and Relu function; however, with the Tanh and Sigmoid activation functions these performances tend to increase when the neurons increase as shown in the figure 6h and figure 6i.

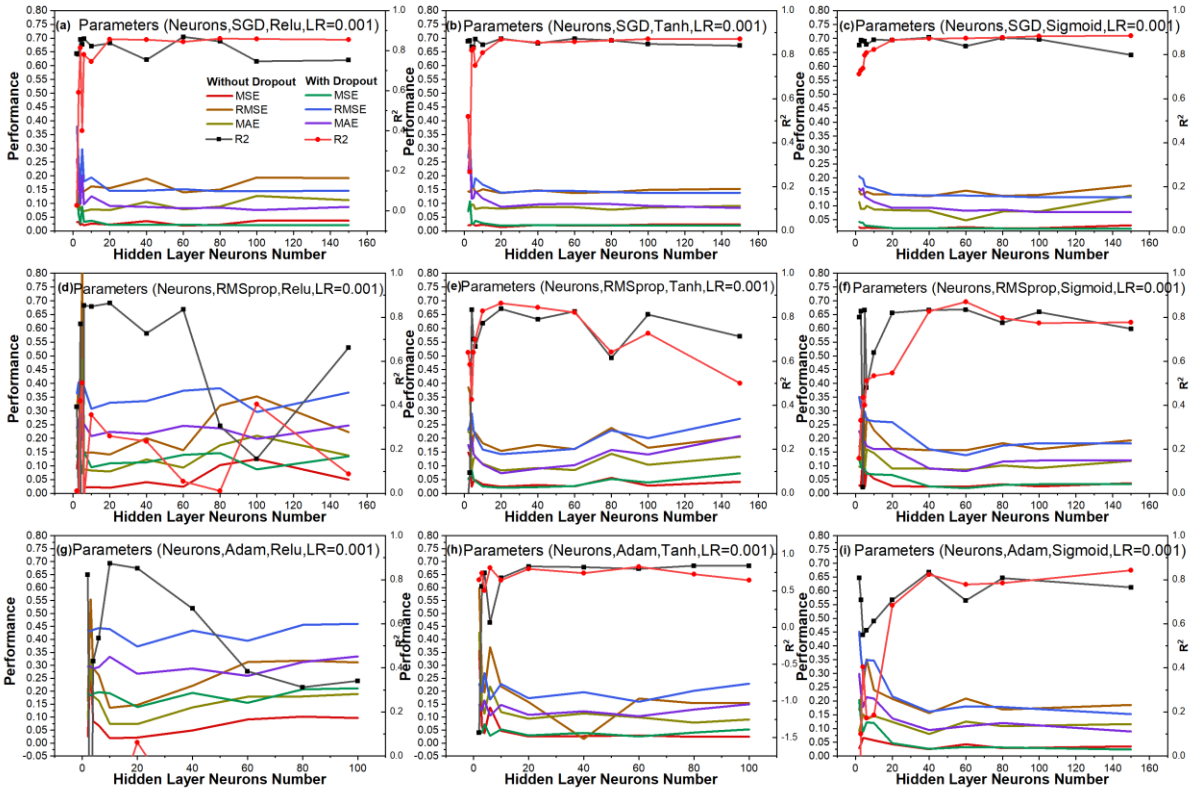

Figure 6. Error metrics for different architectures with SGD, RMSprop, Adam optimizer and Relu, Tanh and Sigmoid activation functions to different cases with and without neuron dropout.

4.4. Solar radiation forecast

Table 2 and table 3 show error metrics values of architectures evaluated with and without neuron dropout.

Table 2 shows that the SGD optimizers with different activation functions have low MSE, RMSE and MAE, and a high value of R^2 compared to the other architectures. It is also observed that SGD and Sigmoid present an R^2 with a value of 0.87761 greater than SGD using Tanh and Relu functions.

Table 3 shows different architectures with neuron dropout. These have high R^2 when neurons are increased. It is also observed that the architecture (150, SGD, Sigmoid, 0.2) have the highest R^2 among all architectures evaluated with and without neuron dropout.

Table 2. Error metrics for different architectures without neuron dropout.

MLP architectures	MSE	RMSE	MAE	R^2
(60, SGD, Relu)	0.01948	0.13956	0.07768	0.86782
(60, SGD, Tanh)	0.01885	0.13729	0.08561	0.87209
(40, SGD, Sigmoid)	0.01804	0.13429	0.08184	0.87761
(20, RMSprop, Relu)	0.02006	0.14162	0.07905	0.86388
(20, RMSprop, Tanh)	0.02378	0.15419	0.08369	0.83865
(60, RMSprop, Sigmoid)	0.02439	0.15618	0.08599	0.83447
(10, Adam, Relu)	0.01859	0.13636	0.07527	0.87382
(80, Adam, Tanh)	0.02371	0.15399	0.07855	0.83908
(40, Adam, Sigmoid)	0.02443	0.15631	0.08010	0.83419

Table 3. Error metrics for the different architectures with neuron dropout.

MLP architectures	MSE	RMSE	MAE	R^2
(80, SGD, Relu, 0.2)	0.02063	0.14364	0.08479	0.85998
(100, SGD, Tanh, 0.2)	0.01892	0.13755	0.09041	0.87161
(150, SGD, Sigmoid, 0.2)	0.01697	0.13026	0.07744	0.88484
(100, RMSprop, Relu, 0.2)	0.08752	0.29584	0.19751	0.40602
(20, RMSprop, Tanh, 0.2)	0.02015	0.14196	0.07285	0.86324
(60, RMSprop, Sigmoid, 0.2)	0.01907	0.13808	0.08115	0.87061
(20, Adam, Relu, 0.2)	0.13837	0.37198	0.26685	0.06097
(60, Adam, Tanh, 0.2)	0.02527	0.15898	0.10314	0.82848
(150, Adam, Sigmoid, 0.2)	0.02311	0.15200	0.08946	0.84320

The best architectures of the model are used to train with the entire time series of solar radiation and make forecasts for the next 6 days, this is shown in figure 7. These architectures are taken from table 1 and 2, which show the highest performance among all the architectures evaluated. The thick lines of figure 7 show the architectures predictions on the test data without neuron dropout and the thin lines of the same color show the architectures predictions for the next 6 days. Similarly, the dotted lines describe the architectures predictions with neuron dropout and the dotted lines of the same faint color are the forecast for the same following days.

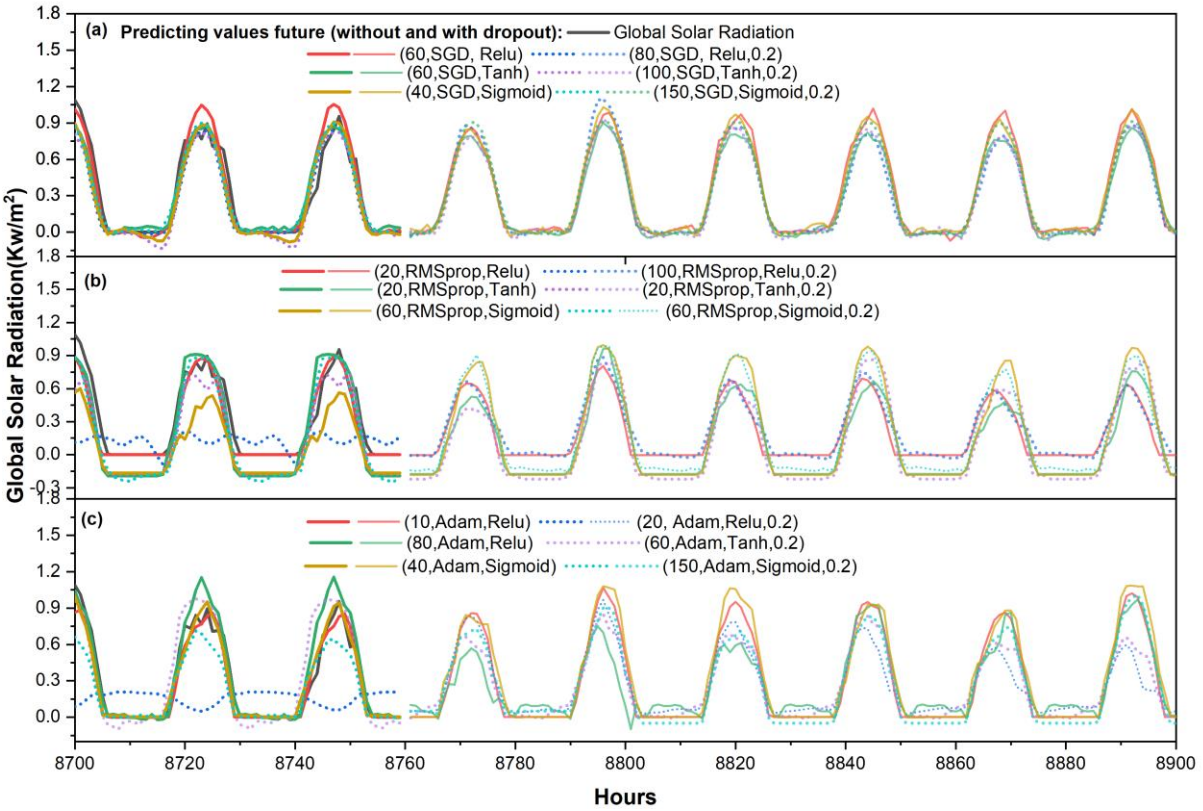

Figure 7. Forecast of global solar radiation for different architectures with SGD, RMSprop and Adam optimizers and their activation functions (Relu, Tanh and Sigmoid).

5. Conclusion

Solar radiation forecasting allows photovoltaic generation to be safe and reliable for an isolated or grid-connected hybrid generation system. Knowing the solar radiation in the future will allows the operators of these systems to give a greater percentage of participation to photovoltaic generation, reducing the operating costs, technical and environmental effects of the system.

In this article, the behavior of MLP was studied for its different architectures in relation to its 4 hyperparameters (optimizer, activation function, number of neurons in the hidden layer, neuron dropout) to predict the global solar radiation time series and their future values.

In the training and validation curves were found that when neuron dropout is added (dropout = 0.2) in the hidden layer, these curves tend to separate and their separation distances decrease or increase when their hyperparameters vary (number of neurons, type activation function and optimizer). The importance of introducing neuron dropout is to avoid overfitting the model and to have better performance. It was also found in the validation curves that the SGD optimizer with the three activation functions had very little peaks or noises, which indicates a good learning of the model. This good performance of the MLP model using the SGD optimizer can be seen in error metrics, predictions on the time series and its realized forecast.

In the performance of architectures without neuron dropout, it was found that the architectures (60, SGD, Sigmoid), (10, Adam, Relu) and (60, SGD, Sigmoid) have an R^2 around 0.877, 0.873 and 0.872, respectively. For other cases, where neuron dropout was considered, the architecture (150, SGD, Sigmoid, 0.2) had a higher performance among all the architectures evaluated, giving an R^2 value of 0.884.

The architectures with the SGD optimizer and three activation functions had more stable predictions in the future predictive values than other architectures.

As future work, it is proposed to use intelligent optimizers that allow to make a great combination of more hyperparameters, find the most optimal MLP architecture, and obtain a higher performance than we found. It is also suggested to work with greater time series data for the analysis, as well as to study their confidence intervals for future values.

Acknowledgments
This work is supported by the Doctoral Program in Sciences with mention in Energy from the National University of Engineering and the Peruvian National Council for Science and Technology (CONCYTEC) through Contract No. 207-2015-Fondecyt-UNI.

References

[1] Zohuri B 2020 *Nuclear fuel cycle and decommissioning* (Albuquerque, NM, United States: Elsevier Ltd.)
[2] Das U K, Tey K S, Seyedmahmoudian M, Mekhilef S, Idris M Y I, Van Deventer W, Horan B and Stojcevski A 2018 Forecasting of photovoltaic power generation and model optimization: A review *Renew. Sustain. Energy Rev.* **81** 912–28
[3] Papadis E and Tsatsaronis G 2020 Challenges in the decarbonization of the energy sector *Energy* **205** 118025
[4] Kannan N and Vakeesan D 2016 Solar energy for future world: - A review *Renew. Sustain. Energy Rev.* **62** 1092–105
[5] IRENA 2019 *Future of solar photovoltaic: Deployment, investment, technology, grid integration and socio-economic aspects* vol November
[6] Yin J, Molini A and Porporato A 2020 Impacts of solar intermittency on future photovoltaic reliability *Nat. Commun.* **11** 1–9
[7] Blanco H and Faaij A 2018 A review at the role of storage in energy systems with a focus on Power to Gas and long-term storage *Renew. Sustain. Energy Rev.* **81** 1049–86
[8] Trevizan R D, Headley A J, Geer R, Atcitty S and Gyuk I 2021 Integration of energy storage with diesel generation in remote communities *MRS Energy Sustain.* **8** 57–74
[9] Sengupta M, Habte A, Stoffel T, Perez R, Myers D, Gueymard C and Philippe Blanc 6 and Stefan Wilbert 2017 *Best Practices Handbook for the Collection and Use of Solar Resource Data for Solar Energy Applications: Second Edition*
[10] Shadab A, Said S and Ahmad S 2019 Box–Jenkins multiplicative ARIMA modeling for prediction of solar radiation: a case study *Int. J. Energy Water Resour.* **3** 305–18
[11] Nespoli A, Ogliari E, Leva S, Pavan A M, Mellit A, Lughi V and Dolara A 2019 Day-ahead photovoltaic forecasting: A comparison of the most effective techniques *Energies* **12** 1–15
[12] Inman R H, Pedro H T C and Coimbra C F M 2013 Solar forecasting methods for renewable energy integration *Prog. Energy Combust. Sci.* **39** 535–76
[13] Ristow D C M, Henning E, Kalbusch A and Petersen C E 2021 Models for forecasting water demand using time series analysis: a case study in Southern Brazil *J. Water, Sanit. Hyg. Dev.* **11** 231–40
[14] Lima S, Gonçalves A M and Costa M 2019 Time series forecasting using Holt-Winters exponential smoothing: An application to economic data *AIP Conf. Proc.* **2186**
[15] Voyant C, Notton G, Darras C, Fouilloy A and Motte F 2017 Uncertainties in global radiation time series forecasting using machine learning: The multilayer perceptron case *Energy* **125** 248–57
[16] Meenu Sreedharan, Khedr A M and El Bannany M 2021 A Multi-Layer Perceptron Approach to Financial Distress Prediction with Genetic Algorithm *Autom. Control Comput. Sci. 2020 546* **54** 475–82
[17] Jain Y K and Bhandare S K 2013 Min Max Normalization Based Data Perturbation Method for Privacy Protection *Int. J. Comput. Commun. Technol.* **4** 233–8
[18] Vafaeipour M, Rahbari O, Rosen M A, Fazelpour F and Ansarirad P 2014 Application of sliding

window technique for prediction of wind velocity time series *Int. J. Energy Environ. Eng.* **5** 1–7
[19] Sugiartawan P, Pulungan R and Kartika A 2017 Prediction by a Hybrid of Wavelet Transform and Long-Short-Term-Memory Neural Network *Int. J. Adv. Comput. Sci. Appl.* **8**
[20] Serin G, Sener B, Ugur Gudelek M, Ozbayoglu A M and Unver H O 2020 Deep multi-layer perceptron based prediction of energy efficiency and surface quality for milling in the era of sustainability and big data *Procedia Manuf.* **51** 1166–77
[21] Matindife L, Sun Y and Wang Z 2020 A Machine-Learning Based Nonintrusive Smart Home Appliance Status Recognition *Math. Probl. Eng.* **2020**
[22] Gardner M W and Dorling S R 1998 Artificial neural networks (the multilayer perceptron) - a review of applications in the atmospheric sciences *Atmos. Environ.* **32** 2627–36
[23] Alboaneen D A, Tianfield H and Zhang Y 2017 Glowworm swarm optimisation for training multi-layer perceptrons *BDCAT 2017 - Proc. 4th IEEE/ACM Int. Conf. Big Data Comput. Appl. Technol.* 131–8
[24] Mas J F and Flores J J 2008 The application of artificial neural networks to the analysis of remotely sensed data *Int. J. Remote Sens.* **29** 617–63
[25] Banerjee C, Mukherjee T and Pasiliao E 2020 The Multi-phase ReLU Activation Function *ACMSE 2020 - Proc. 2020 ACM Southeast Conf.* 239–42
[26] Nwankpa C E 2020 Advances in optimisation algorithms and techniques for deep learning *Adv. Sci. Technol. Eng. Syst.* **5** 563–77
[27] Khan M and Noor S 2019 Performance Analysis of Regression-Machine Learning Algorithms for Predication of Runoff Time *Agrotechnology* **08** 1–12
[28] Watt N and du Plessis M C 2020 Dropout for Recurrent Neural Networks (Springer, Cham) pp 38–47